AMENDMENTS TO THE SPECIFICATION:

Please replace the paragraph beginning on page 11, line 4 with the following amended paragraph:

When a monitored process is running normally, the monitored process may operate with the FIFO communications link open for a write operation. The process for monitoring a process may be to post a blocking read on the monitored process' FIFO[[.]] communications link. A "blocking read" refers to a read I/O operation on the FIFO that causes the thread of execution in the monitoring process that posts the blocking read to pause until, for example, one of two situations occur. These situations may be, for example, data is available to be read from the FIFO or the process at the other end of the FIFO closes its end of the FIFO. It may be the second of these cases that is used to determine that the monitoring process has terminated. In such a situation, the FIFO may be closed on process termination in a manner that is detectable by the monitoring process independently of how the monitored process terminates. This happens, for example, because on abnormal termination, the operating system itself, for example, UNIX, will ensure that the monitored process' end of the FIFO is closed. There may be no actual data transmission occurring between the two processes across the FIFO link. It may be the closing of the channel that is used to determine that the monitored process has terminated.

Please replace the paragraph beginning on page 13, line 13 with the following amended paragraph:

In **Figure 3A**, process **302**, process **304** and process **306** form a process ring known as the watchdog set. Process **302** monitors process **304**, process **304** monitors process **306** and process **306** monitors process **302**. Each process **302-306**, monitors IPC **310** when a new process, such as, process **308**, requests to join the watchdog set. IPC **310** is the gateway in which a new process joins the ring. The IPC channel is monitored by all processes within the watchdog set. This ensures that no one process is considered the "master" monitor, for example, the process to which join requests would otherwise

need to be sent. When a process requests to join the watchdog set, it sends a message on this IPC channel. Any of the processes monitoring this IPC channel may read this message (typically chosen at random by the operating system). Therefore, it may be essential that the message be able to be read atomically, i.e., in a single operation, so the message does not get split into parts and removing the possibility that the monitoring processes could see fragmented messages. This is achieved [[the]] in the example implementation by choosing messages one (1) byte in length. Process **308** requests, through IPC **310**, to join the watchdog set.

Please replace the paragraph beginning on page 17, line 31 with the following amended paragraph:

This mechanism may be provided as a FIFO, called the join FIFO. Each running privileged process may have a thread reading from the join FIFO, called the join thread. On start up, for example, such a privileged process may open the join FIFO for a write operation in a non-blocking mode. If the open operation succeeds, this may indicate that there is at least one reader process, i.e., at least one such privileged process.

Please replace the paragraph beginning on page 23, line 18 with the following amended paragraph:

If the termination of the process within the watchdog set was not a normal termination (step **514:NO**), an attempt is made to restart the terminated monitored process (step **518**). [[then]] Then a privileged restart state is initiated (step **520**). A determination is then made as to whether or not the terminated monitored process was restarted (step **522**). If the terminated monitored process was not restarted (step **522:NO**), the operation returns to step **518** in which an attempt is made to restart the terminated monitored process. If the terminated monitored process was restarted (step **522:YES**), the operation returns to step **516** in which monitoring assignments are reassigned.

Please replace the paragraph beginning on page 24, line 1 with the following amended paragraph:

Figure 6 is an exemplary flowchart illustrating the initialization of processes that will participate in the watchdog set in accordance with a preferred embodiment of the present invention. In this example, the operation starts by reading the state file (step **600**) and determining as to whether or not the state is a privileged start or a privileged restart (step **602**). If the state is not a privileged start nor a privileged restart (step **602:NO**), then a determination is made as to whether or not a join FIFO can be opened or read (step **604**). If the join FIFO can be opened or read (step **604:YES**), then a message is sent on the join FIFO request to be started (step **616**). The process will then be restarted on a privileged restart state by one of the processes currently running (step **618**) and thereafter the operation terminates. If the join FIFO cannot be opened or read (step **604:NO**), a process is designated as the first process (step **606**). A special privilege is obtained if necessary (step **608**). The first process is recorded so that any child started should be instructed to monitor the first process (step **610**). Then a determination is made as to whether or not there is another process that should be started (step **612**). If there is another process that should be started (step **612:YES**), one of the processes required to be started are chosen (step **626**). The chosen process' state is set to indicate a "privilege start" state (step **624**). The chosen process is instructed to monitor the appropriate process (step **622**). The chosen process is started (step **620**) and the operation continues to step **642** in which a monitoring thread is started to monitor the process.

Please replace the paragraph beginning on page 26, line 5 with the following amended paragraph:

Figure 7 is an exemplary flowchart illustrating handling of a request to join a watchdog set in accordance with a preferred embodiment of the present invention. In this example, the operation starts by reading a join FIFO for join requests (step **702**). Then a determination is made as to whether or not the join request was received (step **704**). If the join request was not received (step **704:NO**), the operation returns to step **702** in

which the join FIFO are read for join requests. If the join request was received (step **704:YES**), a requesting process' state is set to "privileged start" (step **706**). A determination is then made as to whether or not only one process is running (step **708**). If only [[on]] one process is running (step **708:YES**), a new process is instructed to monitor only the running process (step **710**) and thereafter the operation continues to step **716** in which a new process is started as a child. If more than one process is running (step **708:NO**), a new process is instructed to monitor the currently running process (step **712**). A monitoring process is stopped which is currently being monitored (step **714**). Then the new process is started as a child (step **716**). A monitoring thread is started to monitor the new process (step **718**) and thereafter the operation returns to step **702** in which a join FIFO is read for join requests.

Please replace the paragraph beginning on page 26, line 30 with the following amended paragraph:

The advantages provided by the present invention should be apparent in view of the detailed description of the invention provided above. The ability to dynamically join and leave the watchdog set and the absence of the requirement of any one process having special significance of acting as an overall monitor over the watchdog set is avoided. Therefore, the present invention provides an advantage that other solutions to the problem of monitoring processes within a set do not address. However, any operating system may be used to ~~implemented~~ implement the operation of the present invention. This product also uses, for example, disk files for storing state and process monitoring information as described above, although any type of storage device may be used to store the state and process monitoring information, such as, for example, a hard disk, a magnetic tape, and the like. An advantage of the present invention is to ensure a high availability of the processes used to provide, for example, enhanced security to an operating system, such as, for example, a UNIX operating system.

Please replace the paragraph beginning on page 27, line 19 with the following amended paragraph:

The present invention provides a method in which a process within a set of processes monitors another process within the set. In addition, each monitoring process is monitored. If a process leaves the set of processes in a normal manner, no further action is taken toward the exiting process. However, monitoring assignments of the processes remaining in the set are reassigned so that all processes [[area]] are again monitored. However, if a process within the set of processes leaves the set in an abnormal manner, the process which is monitoring the exiting process will attempt to restart the process to bring it back into the set. Furthermore, if a process is not originally in the set, a mechanism is provided for bringing the new process into the set of processes. Once entry into the watchdog set is accomplished by the new process, monitoring assignments are established such that a chosen process within the group begins to monitor the new process and the new process begins monitoring the process formally monitored by the chosen process. "Randomness" occurs in that one of the already running processes reading on the join FIFO will be randomly chosen by an operating system to receive the request to join the watchdog set. Which ever process reads this request will start and monitor the requesting process. Prior to being started, the requesting process will be instructed to monitor whichever process that the process handling the join request was previously monitoring.